4

Classification and Reformulation

Motivation

Given the diversity of planning functions in the supply chain planning matrix described in Chapter 2, and given the diversity of supply chains (each supply chain can be characterized by a combination of functional and structural attributes, implying a huge diversity in planning requirements; see Section 2.3), a single advanced planning system or a single monolithic mathematical programming planning model cannot represent all planning problems.

Therefore, in parallel to the supply chain typology, our approach for the construction of planning models is to decompose and classify them based on their main attributes: decisions, objectives, and constraints. This building block approach and classification helps us and allows us first to construct a model and an initial mathematical formulation for the planning problem to be addressed.

Beyond modeling, there is a second and major motivation for this classification. Most real-life production planning problems are complex because they involve many products and many resources, such as machines, storage facilities, and plants, and many restrictions have to be satisfied by acceptable production plans. This results in mixed integer programs of large size that are usually very difficult to solve.

In Chapter 3 we have surveyed the state-of-the-art generic branch-and-bound and branch-and-cut algorithms based on a priori reformulation, valid inequalities, and separation. In the literature many reformulation results are known and described for canonical production planning models, such as single-item and/or single-resource problems, which are much simpler than the complex real-life problems.

In order to be able to incorporate these reformulation results in specialized branch-and-bound/cut algorithms for solving production planning models, it is crucial to be able to identify which results to use, which requires one to

identify which canonical submodels are present in a model. The classification scheme presented here pursues exactly this goal.

Objective

In the context of a decomposition approach, it is the specific objective of this chapter to

- describe and classify the *canonical production models* frequently occurring as relaxations or sub-models in real life production planning problems,
- identify and classify the *reformulation results* that are known for these canonical models in order to design efficient branch-and-bound/cut algorithms for solving practical production planning models.

For complex planning problems, the objective is also to present and illustrate the effectiveness of a systematic reformulation procedure allowing us to take advantage – through the classification scheme – of reformulation results for standard single-item subproblems to obtain improved formulations and to design branch-and-cut optimization algorithms.

In Chapter 5 we then demonstrate how to use the classification scheme and the reformulation procedure in practice, with appropriate software tools.

The (more) technical and detailed presentation and derivation of the reformulation results listed here, as well as some additional reformulation results and techniques useful for more complex models (but requiring a less automatic approach) are given in Parts II to IV, and illustrated in Part V.

Contents

Step by step:

- In Section 4.1 we illustrate on the *LS-U* (uncapacitated lot-sizing) production planning model the use and impact of reformulations on the performance of the branch-and-bound/cut algorithm, namely the effect of using the extended (or compact linear) reformulation technique and the cutting plane reformulation technique defined in Chapter 3.
- In Section 4.2 we describe the *decomposition approach* used to reformulate and solve complex planning models involving many items and resources, starting from available reformulations for simpler (i.e., single-item, single-machine) planning models.
- In Section 4.3 we describe our *classification scheme* for canonical single item production planning models in the form of a three-field identifier $PROB\text{-}CAP\text{-}VAR$ for each model, and by giving a conceptual or verbal description as well as an initial mathematical formulation for each model.

- In Section 4.4 we describe a *systematic reformulation procedure* relying on tables of extended and cutting plane reformulation results for the most common single-item production planning models, including *LS-U*.
- In Section 4.5 we put together these ideas to illustrate the use and effectiveness of the systematic reformulation procedure on the Master Production Scheduling example from Section 1.2.

4.1 Using Reformulations for Lot-Sizing Models

In earlier chapters we have presented and illustrated the modeling and optimization approach, as well as the generic branch-and-bound and branch-and-cut algorithms used to solve the resulting models.

Model *LS-U* is the simplest high-level relaxation occurring in most production planning models. So, finding good reformulations for *LS-U* is an important first step. Here we use this model to illustrate the type of reformulation results available for canonical single-item planning models, namely a priori reformulations and cutting planes with separation.

The approach is illustrated on the first *LS-U* example described in Section 1.1. For simplicity, we recall here the initial formulation of this *LS-U* instance characterized by

- the demand satisfaction constraint for the single product *bike* over eight consecutive time periods,
- the variable upper bound constraint for the single product *bike* over eight consecutive time periods, and
- the initial inventory of product *bike*.

$$\min \quad cost \ := \sum_{t=1}^{NT} (p\ x_t + q\ y_t) \ + \sum_{t=1}^{NT-1} h\ s_t \tag{4.1}$$

$$dem\_sat_t \ := \quad s_{t-1} + x_t = d_t + s_t \qquad \text{for } 1 \le t \le NT \tag{4.2}$$

$$s_0 = s\_ini,\ s_{NT} = 0 \tag{4.3}$$

$$vub_t \ := \quad x_t \le (\sum_{k=t}^{NT} d_k) y_t \qquad \text{for } 1 \le t \le NT \tag{4.4}$$

$$x_t, s_t \in \mathbb{R}_+,\ y_t \in \{0,1\} \qquad \text{for } 1 \le t \le NT, \tag{4.5}$$

where the variables are x_t for production, s_t for inventory, and y_t for set-up in period t, and the data are $NT = 8$, $p = 100$, $q = 5000$, $h = 5$, $d = [400, 400, 800, 800, 1200, 1200, 1200, 1200]$ and $s\_ini = 200$. This formulation is $O(NT) \times O(NT)$; that is, it involves on the order of NT constraints and NT variables (see Section 3.2), where NT represents the number of time periods in the planning horizon.

The performance of the branch-and-bound algorithm (using the default Xpress-MP optimizer, but without preprocessing and without using the cuts generated by the solver) on this initial formulation (4.1)–(4.5) is reported in Table 4.1.

Table 4.1. B&B Solution of the *LS-U* Example from Section 1.1

Formulation Size Algorithm	LP Val. Vars Cons	CPLP Val. CPLP Time CPLP Cuts	OPT Val. OPT Time OPT Nodes
(4.1)–(4.5)	712,189	–	736,000
$O(NT) \times O(NT)$	24	–	0
B & B	16	–	29

In Table 4.1, "Vars" and "Cons" represent the number of variables and constraints in the formulation, "LP Val." is the value of the initial linear relaxation of the formulation. "CPLP Val.", "CPLP Time" and "CPLP Cuts" give, respectively, the value of the lower bound at the root node after the addition of cutting planes, the cutting plane time at the root node, and the number of cuts in the formulation at the end of the root node. "CPLP" values are only reported for branch-and-cut algorithms (the Xpress-MP cuts are not used in this toy example). "OPT Val.", "OPT Time" and "OPT Nodes" are the value of the optimal solution, the total run-time and total number of nodes in the enumeration tree. Times are given in seconds, rounded to the nearest integer.

In our analysis, we concentrate on the lower bound value at the root node and on the total number of nodes in the enumeration tree. Both indicators measure the quality of the formulation used. The run-time (rounded to 0 second) and the gap (always 0 when an optimal solution is found) do not give much information in this tiny example.

Observation 4.1 *There are 29 branch-and-bound nodes with the initial formulation. Observe that this formulation is already using some tightening for the variable upper bound constraint (4.4). If this constraint is replaced by the simpler but usual big-M type constraint* $x_t \leq My_t$*, with* $M = 10,000$*, that is, if we do not introduce the tightest upper bound on* x_t *in (4.4), then the LP lower bound at the root node ("LP Val.") is reduced to 703,500 and the number of nodes needed to solve the model to optimality increases to 51 nodes. So, some straightforward a priori formulation tightening is already included in the initial model.*

4.1.1 Using A Priori Extended Reformulations

As explained in Section 3.4, we look now for tight reformulations of *LS-U*, or tight reformulations of high-level relaxations of *LS-U*.

LS-U is polynomially solvable as it can be solved by dynamic programming. Given the complexity equivalence between optimization and separation discussed in Part II, it is natural to look for a compact (i.e., polynomial in the number of variables and constraints) linear reformulation for *LS-U*. As an example, we describe and test here a well-known extended reformulation for *LS-U*.

Multi-Commodity Extended Reformulation

A classical way to tighten the formulation of fixed charge network flow problems is to decompose the flow along each arc of the network as a function of its destination. This defines a so-called multi-commodity formulation by assigning a different commodity to each destination node. The decomposition by commodity allows one to tighten the formulation by decreasing the upper bounds in the variable upper bound constraints, which is important as illustrated in Observation 4.1.

We have already given the network flow interpretation of *LS-U* in Figure 1.2 in Section 1.1. So we can apply the multi-commodity idea, and decompose the flow (production) x_t as a function of its destination node (demand period) t, $t+1$, ..., NT. Similarly, we can decompose the flow (inventory) s_t as a function of its destination node (demand period) $t+1$, $t+2$, ..., NT.

So, we consider as one specific commodity the demand to be satisfied in each time period, and do not mix the commodities. Commodity t corresponds to the demand delivered in period t. We define the new variables x_{it} $(i \leq t)$ as the production in period i of commodity t, and the new variable s_{it} $(i < t)$ as the inventory at the end of period i of commodity t.

In this reformulation, we further constrain the initial inventory s_ini to be consumed in the first period; that is, $s_{01} = s\_ini$ and $s_{0t} = 0$ for $t = 2, \ldots, NT$. This can be done without loss of optimality, because $s\_ini \leq d_1$ and there is always an optimal solution where earlier production is delivered first (this is called FIFO [= First In First Out] or FPFD [= First Produced First Delivered] ordering).

Also, the variables s_{tt}, for $t = 1, \ldots, NT$, do not exist because commodity t must be delivered in period t, therefore no inventory of commodity t may exist at the end of period t.

If needed, for instance, in case of positive minimal stock at the end of the planning horizon, an additional commodity can be created to correspond to the end horizon inventory. This is not necessary here, as we assume that the stock at the end of the horizon is zero.

The flow conservation constraint obtained for commodity $t = 5$ is illustrated in Figure 4.1.

By modeling separately the demand satisfaction (flow conservation) for each commodity, the *LS-U* model (4.1)–(4.5) can be reformulated as

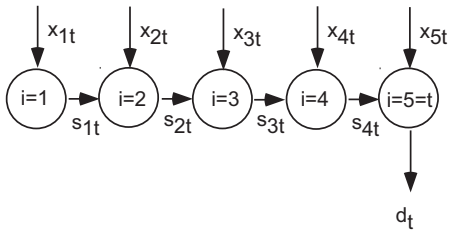

Figure 4.1. The flow conservation global constraint for commodity $t = 5$.

$$\min \quad cost := \sum_{i=1}^{NT} \sum_{t=i}^{NT} (p\ x_{it} + h\ s_{it}) \ + \ \sum_{i=1}^{NT} q\ y_i \tag{4.6}$$

$$dem\_sat_{it} := \quad s_{i-1,t} + x_{it} = \delta_{it} d_t + s_{it} \qquad \text{for } 1 \le i \le t \le NT \tag{4.7}$$

$$s_{01} = s\_ini,\ s_{0t} = 0 \qquad \text{for } 2 \le t \le NT \tag{4.8}$$

$$s_{tt} = 0 \qquad \text{for } 1 \le t \le NT \tag{4.9}$$

$$vub_{it} := \quad x_{it} \le \hat{d}_t y_i \qquad \text{for } 1 \le i \le t \le NT \tag{4.10}$$

$$s_{it}, x_{it} \in \mathbb{R}_+,\ y_i \in \{0,1\} \qquad \text{for } 1 \le i \le t \le NT, \tag{4.11}$$

where the notation δ_{it} denotes 1 if $i = t$, and 0 otherwise. Constraint (4.7) is the flow conservation constraint of commodity t in all periods $i = 1, \ldots, t$, where the only period i with a demand for commodity t is $i = t$. Constraints (4.8) and (4.9) impose that there is no initial and no final inventory (end of period t) of commodity t, except the initial inventory of commodity 1. Constraint (4.10) forces the set-up variable y_i to be 1 when there is production for commodity t in period i. Using the decomposition of the flow, the tightest upper bound on x_{it} is $\hat{d}_t$, where $\hat{d}_1 = d_1 - s\_ini$ and $\hat{d}_t = d_t$ for $t > 1$. Constraint (4.11) imposes the nonnegativity and binary restrictions on the variables. Finally, Constraint (4.6) expresses the cost of the production plan.

This extended reformulation does not contain the initial variables and constraints. But it is nevertheless a valid reformulation of *LS-U* in the sense of Definitions 3.3 and 3.13. This can observed because an equivalent reformulation to (4.6)–(4.11) would be obtained by adding constraints to define the initial variables as a function of the new decomposed variables (i.e., $x_i = \sum_{t=i}^{NT} x_{it}$ and $s_i = \sum_{t=i+1}^{NT} s_{it}$ for all i), and by keeping the original objective function (4.1). The reformulation has then the same feasible solutions in the original (x, s, y) space as the original model.

Testing the Multi-Commodity Extended Reformulation

Reformulation (4.6)–(4.11) is of size $O(NT^2) \times O(NT^2)$. We can now test the effectiveness of the decomposition by commodity and tightening of the

variable upper-bound constraints. The performance of the branch-and-bound algorithm (again using the default Xpress-MP optimizer, but without preprocessing and without the cuts generated by the solver) on the initial formulation and on the multi-commodity reformulation are compared in Table 4.2.

Table 4.2. B&B Solution of the *LS-U* Example from Section 1.1, Comparison of Initial and Multi-Commodity Formulations

Formulation Size Algorithm	LP Val. Vars Cons	CPLP Val. CPLP Time CPLP Cuts	OPT Val. OPT Time OPT Nodes
(4.1)–(4.5)	712189	–	736,000
$O(NT) \times O(NT)$	24	–	0
B & B	16	–	29
(4.6)–(4.11)	736,000	–	736,000
$O(NT^2) \times O(NT^2)$	72	–	0
B & B	72	–	1

We observe in Table 4.2 that the multi-commodity reformulation solves *LS-U* without any branching. The LP value is the optimal value, and one node suffices. The following theorem shows that this is not chance. The multi-commodity reformulation solves all instances of *LS-U* without branching.

Theorem 4.1 *The linear relaxation of formulation ((4.6)–(4.11)) always has an optimal solution with y integer, and solves LS-U. In other words, formulation (4.7)–(4.11) is a tight extended formulation of the convex hull of feasible solutions to LS-U. This is also called a complete linear description of LS-U.*

The multi-commodity extended reformulation has been given here to illustrate the type of results one can obtain with reformulations. Other extended reformulations giving a complete linear description of *LS-U* are known, as well as extended reformulations for canonical models other than *LS-U*. Pointers to these extended formulations are defined in our systematic reformulation procedure in Section 4.4.

4.1.2 Using Cutting Planes

We have just described the multi-commodity reformulation for the single-item model *LS-U* with NT periods. Although this reformulation is as tight as possible, it has $O(NT^2)$ constraints and $O(NT^2)$ variables and a model with 32 time periods has over a thousand variables and a thousand constraints. This may be too large a reformulation if it has to be applied to all items in a large-size multi-item production planning model (see Section 4.2).

One way to overcome this difficulty is to look for a complete linear description of model *LS-U* in the initial variable space involving only $O(NT)$

variables. And if this complete linear description needs an exponential (in NT) number of constraints, we can use the cutting plane and separation approach described in Section 3.5 to avoid adding all these constraints a priori.

A Class of Valid Inequalities

A first class of valid inequalities can be easily identified from the fractional solution of the linear relaxation of the initial formulation. Figure 4.2 represents the optimal solution of the linear relaxation of (4.1)–(4.5), where missing arcs correspond to arcs with zero flow.

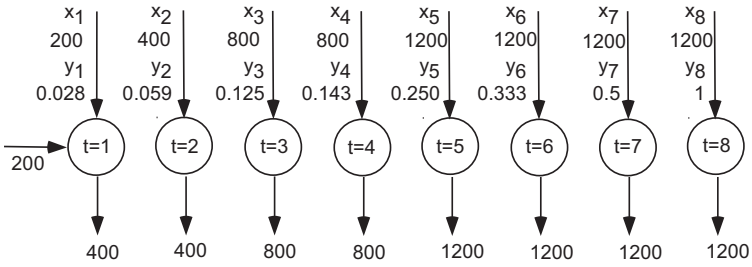

Figure 4.2. The solution of the linear relaxation of (4.1)–(4.5).

To eliminate or cut off this fractional solution we have to look at periods in which the corresponding y variable is fractional. Consider period 2 with $x_2 = 400$ but $y_2 = 0.059$. This value of y_2 is minimized because of the objective function, therefore y_2 is exactly the minimal value allowed by the set-up forcing constraint (4.4); that is, $y_2 \geq (=) \frac{x_2}{d_{2,8}}$ where the notation $d_{\alpha\beta}$ denotes $\sum_{i=\alpha}^{\beta} d_i$.

Observe that this reasoning also applies to y_8, but y_8 takes the value 1 because $y_8 \geq (=) x_8/d_{8,8} = 1$.

So, if period 2 was the last period of the horizon, we could also write $x_2 \leq d_2 y_2$ as set-up forcing constraint, and y_2 would take the value 1 with the current value of x_2.

But d_2 is a valid upper bound on x_2 only if period 2 is the last period or if there is no stock at the end of period 2 (i.e., $s_2 = 0$ and period 2 is separated from the later periods as if period 2 were the last one). Hence a valid upper bound on x_2 is $x_2 \leq d_2 + s_2$.
Therefore, a logical implication is

$$s_2 = 0 \;\Rightarrow\; x_2 \leq d_2 y_2.$$

This implication can be converted into the valid linear inequality

$$x_2 \leq d_2 y_2 + s_2 \ ,$$

which is valid because in any feasible solution to *LS-U*:

- Either $y_2 = 0$ and the inequality is satisfied because $y_2 = 0$ implies $x_2 = 0$, and $0 \leq s_2$;
- Or $y_2 = 1$ and $x_2 \leq d_2 + s_2$ is a valid upper bound on x_2.

This inequality is violated by the current fractional point from Figure 4.2, and so we have simulated one pass of the separation problem. Using the same reasoning and starting from the upper bound $x_t \leq d_{tl} + s_l$ for any $l \geq t$ (remember that d_{tl} denotes $\sum_{i=t}^{l} d_i$), the above valid inequality can easily be generalized to

$$x_t \;\leq\; d_{tl} y_t \;+\; s_l \quad \text{for all } 1 \leq t \leq l \leq NT \tag{4.12}$$

for arbitrary demand data and time period.

Complete Linear Description

We denote the set of feasible solutions of model *LS-U*, that is, the solutions of (4.2)–(4.5), by X^{LS-U}. The class (4.12) of valid inequalities does not suffice to obtain a linear description of $conv(X^{LS-U})$ for any instance.

First we define the more general class of so-called (l, S) inequalities. It is shown in Chapter 7, Proposition 7.4, that the inequalities

$$\sum_{i \in S} x_i \;\leq\; \sum_{i \in S} d_{il} y_i \;+\; s_l \qquad \text{for all } 1 \leq l \leq NT \;\text{ and } S \subseteq \{1, \ldots, l\} \tag{4.13}$$

are valid for X^{LS-U}. As an example, the valid inequality $x_2 + x_3 \leq d_{24} y_2 + d_{34} y_3 + s_4$ corresponds to the inequality (4.13) with $l = 4$ and $S = \{2, 3\} \subset \{1, \ldots, 4\}$.

The next theorem simply states that the (l, S) inequalities suffice to obtain the desired complete linear formulation. We can always eliminate the initial inventory, as we did in the multi-commodity formulation, by assuming a FPFD ordering and updating the (residual) demand vector accordingly. So, without loss of generality we assume that $s\_ini = 0$.

Theorem 4.2 *Assuming $d_t \geq 0$ for all t and $s\_ini = 0$, a complete linear description of $conv(X^{LS-U})$ is*

$$s_{t-1} + x_t = d_t + s_t \qquad \text{for } 1 \leq t \leq NT \tag{4.14}$$

$$s_0 = 0\;, \quad s_{NT} = 0 \tag{4.15}$$

$$x_t \leq d_{t,NT}\, y_t \qquad \text{for } 1 \leq t \leq NT \tag{4.16}$$

$$\sum_{i \in S} x_i \;\leq\; \sum_{i \in S} d_{il} y_i \;+\; s_l \qquad \text{for } 1 \leq l \leq NT,\; S \subseteq \{1, \ldots, l\} \tag{4.17}$$

$$x_t, s_t, y_t \in \mathbb{R}_+,\; y_t \leq 1 \qquad \text{for } 1 \leq t \leq NT\;. \tag{4.18}$$

Note that if $d_1 > 0$, the (l, S) inequality $x_1 \leq d_1 y_1 + s_1$ together with the initial equation $x_1 = d_1 + s_1$ and $y_1 \leq 1$ imply the equality $y_1 = 1$.

Separation Algorithm

We have obtained a complete linear programming formulation of *LS-U* in the original variables (x, s, y). However, this formulation contains an exponential number of (l, S) inequalities (4.13), and a cutting plane approach must be used to avoid adding all these inequalities a priori to the formulation.

In order to use a class of valid inequalities in a cutting plane algorithm, the associated separation problem must be solved. Given a solution to the linear relaxation, it consists of either finding an inequality from the class violated by the solution, or proving that all inequalities from the class are satisfied by the given solution; see Chapter 3.

We denote by P^{LS-U} the initial (linear) formulation (4.2)–(4.4) of *LS-U* together with $x_t, s_t \geq 0$ and $0 \leq y_t \leq 1$ for all t.

Separation Given $(x^*, s^*, y^*) \in P^{LS-U}$:

- Either we find an (l, S) inequality violated by (x^*, s^*, y^*);
- Or we prove that all (l, S) inequalities are satisfied by (x^*, s^*, y^*).

As the (l, S) inequality may be rewritten as $\sum_{i \in S}(x_i - d_{il}y_i) \leq s_l$, to find the most violated (l, S) inequality for fixed $l \in \{1, \ldots, n\}$, it suffices to set

$$S^* \;=\; \{i \in \{1, \ldots, l\} \;:\; (x_i^* - d_{il}y_i^*) > 0 \;\}$$

and test whether $\sum_{i \in S^*}(x_i^* - d_{il}y_i^*) > s_l^*$.

- If this holds, then the (l, S^*) inequality is the most violated inequality for the given value of l.
- Otherwise, there is no violated (l, S) inequality for the given value of l.

By enumerating over all possible values of l, we obtain a separation algorithm for the (l, S) inequalities whose running time is $O(NT^2)$; see Section 3.2.

Testing the Cutting Plane Reformulation

Table 4.3 compares the performance of the three formulations (again using the default Xpress-MP optimizer, without preprocessing or cuts generated by the solver) proposed to solve our bike production planning example from Section 1.1:

- The initial formulation (4.1)–(4.5) solved by branch-and-bound.
- The multi-commodity a priori reformulation (4.6)–(4.11) solved by branch-and-bound.
- The reformulation in the original space of variables using the initial formulation (4.1)–(4.5) and the separation algorithm for the (l, S) inequalities (4.13) in a branch-and-cut or cutting plane algorithm.

Table 4.3. B&B and B&C Solution of the *LS-U* Example from Section 1.1, Comparison of Reformulations

Formulation Size Algorithm	LP Val. Vars Cons	CPLP Val. CPLP Time CPLP Cuts	OPT Val. OPT Time OPT Nodes
(4.1)–(4.5) $O(NT) \times O(NT)$ B & B	712,189 24 16	– – –	736,000 0 29
(4.6)–(4.11) $O(NT^2) \times O(NT^2)$ B & B	736,000 72 72	- – –	736,000 0 1
(4.1)–(4.5) and (4.13) $O(NT) \times O(2^{NT})$ B & C	712,189 24 16	736,000 0 21	736,000 0 1

Our cutting plane algorithm requires six passes (and 21 cuts in total) to solve this instance of *LS-U* without branching, where one pass is defined as one iteration of cut generation for each l with $1 \leq l \leq NT$, followed by a single reoptimization.

4.1.3 Using Approximate Reformulations

The *LS-U* model is an ideal case. We know complete and compact (i.e., polynomial in size) extended linear reformulations, as well as a complete linear description in the original space of the convex hull of solutions $conv(X^{LS-U})$ with a fast separation algorithm. So, when a practical production planning problem involves *LS-U* as a submodel for an item, these reformulations are very effective in improving the formulation.

In many other cases, we only have partial reformulation results for the single-item submodels, say X^{LS}. That is, we have an initial formulation P^{LS}, some extended reformulation, or a class of valid inequalities in the original space that defines only an approximation $\overline{conv}(X^{LS})$ of the convex hull of solutions, but is significantly smaller than the initial formulation; that is,

$$conv(X^{LS}) \subset \overline{conv}(X^{LS}) \subset P^{LS}.$$

These approximate or partial reformulations can be used in the same way – a priori reformulations or cutting planes – as complete reformulations.

In all cases, the objective of the reformulation phase is to be able to use the best known results for submodels embedded in the planning model to be solved. This is the essence of the decomposition approach that we formalize next.

4.2 The Decomposition Approach for Complex Models

As we have already seen in the examples of Chapter 2, and in the master production scheduling example from Section 1.2.3, the structure of many, or most, multi-item production planning problems looks very similar when represented as mixed integer programs.

To be specific, the MPS example is more or less of the form

$$(MIPP^{item}) \qquad W^\star \;=\; \min \textstyle\sum_i \sum_t (p_t^i x_t^i + h_t^i s_t^i + q_t^i y_t^i)$$

$$\begin{array}{lll}
[\ s_{t-1}^i + x_t^i = d_t^i + s_t^i,\ x_t^i \le C_t^i y_t^i,\ y_t^i \le 1 & \text{for all } t\ ], \quad \text{for all } i & (4.19)\\
[\ \displaystyle\sum_i a_t^{ik} x_t^i + \sum_i b_t^{ik} y_t^i \le L_t^k & \text{for all } k\ ], \quad \text{for all } t & (4.20)\\
[\ x_t^i \le C_t^i y_t^i,\ y_t^i \le 1 & \text{for all } i\ ], \quad \text{for all } t & (4.21)\\
x_t^i \in \mathbb{R}_+^1,\ s_t^i \in \mathbb{R}_+^1,\ y_t^i \in \mathbb{Z}_+^1 & \text{for all } i, t. &
\end{array}$$

This can be written more compactly as

$$\begin{array}{lll}
(MIPP^{item}) \quad W^\star \;=\; \min & \displaystyle\sum_i \sum_t (p_t^i x_t^i + h_t^i s_t^i + q_t^i y_t^i) & \\
& (x^i, s^i, y^i) \in Y^i & \text{for all } i\ ,\\
& (x, s, y) \in Z\ , &
\end{array}$$

where Y^i represents the set of feasible solutions to the item i lot-sizing problem (i.e., lot sizes x^i, set-ups y^i, and inventory levels s^i defined for all time periods and satisfying the constraints (4.19) for item i), such as *LS-U* or some of its variants. On the other hand Z represents the solutions satisfying the set of linear constraints (4.20)–(4.21). The constraints defining Z are often called coupling or linking constraints because they link together the items that have to share the joint capacity.

This representation or scheme is not totally general, and certainly not unique. For instance, we can also view the linking set Z as the intersection of independent single-period sets. Now we can write the problem as the intersection of the time and period submodels as in formulation

$$\begin{array}{lll}
(MIPP_{time}^{item}) \quad W^\star \;=\; \min & \displaystyle\sum_i \sum_t (p_t^i x_t^i + h_t^i s_t^i + q_t^i y_t^i) & \\
& (x^i, s^i, y^i) \in Y^i & \text{for all } i\\
& (x_t, s_t, y_t) \in Z_t & \text{for all } t\ ,
\end{array}$$

where Z_t represents the set of feasible solutions to the period t submodel, that is, the lot sizes x_t, set-ups y_t defined for all items, and satisfying the constraints (4.20)–(4.21) for time period t.

The branch-and-bound/cut methods studied in Chapter 3, like most optimization methods, are based on easy-to-solve relaxations of the initial problem. For example, the above problem can be solved by some standard MIP software using a branch-and-bound algorithm based on the linear programming relaxation LR of the initial formulation. We suppose that the initial formulation for the lot-sizing sets Y^i is P^{Y^i}, and the initial formulation for the period t linking constraints in Z_t is P^{Z_t}. So, LR is defined by

$$\begin{aligned} LR \quad = \quad \min \qquad & \sum_i \sum_t (p_t^i x_t^i + h_t^i s_t^i + q_t^i y_t^i) & \\ & (x^i, s^i, y^i) \in P^{Y^i} & \text{for all } i \\ & (x_t, s_t, y_t) \in P^{Z_t} & \text{for all } t. \end{aligned}$$

Unfortunately, this direct branch-and-bound approach can only be used for the solution of small-size problems. In order to solve, or to find good solutions, for more realistic or real-size problems, one has to work with better or tighter relaxations or formulations providing improved lower bounds. Because of the multi-item structure of the initial problem, most efficient solution approaches are based on the following reformulation.

$$\begin{aligned} LB^{item} \quad = \quad \min \qquad & \sum_i \sum_t (p_t^i x_t^i + h_t^i s_t^i + q_t^i y_t^i) & \\ & (x^i, s^i, y^i) \in \overline{conv}(Y^i) & \text{for all } i \\ & (x_t, s_t, y_t) \in P^{Z_t} & \text{for all } t, \end{aligned}$$

where $\overline{conv}(Y^i)$ represents a partial (or complete) reformulation of the convex hull of the solutions of the single-item model Y^i. This bound LB^{item} can be obtained in several ways:

- Either by branch-and-bound using an a priori and compact linear reformulation of $\overline{conv}(Y^i)$;
- Or by branch-and-cut using a reformulation of $\overline{conv}(Y^i)$ involving many constraints, combined with a separation algorithm; see Chapter 3.

In some cases, we may also know good (or complete) linear reformulations for the single-period submodel. This in turn leads us to a stronger linear programming relaxation

$$\begin{aligned} LB_{time}^{item} \quad = \quad \min \qquad & \sum_i \sum_t (p_t^i x_t^i + h_t^i s_t^i + q_t^i y_t^i) & \\ & (x^i, s^i, y^i) \in \overline{conv}(Y^i) & \text{for all } i \\ & (x_t, s_t, y_t) \in \overline{conv}(Z_t) & \text{for all } t, \end{aligned}$$

where $\overline{conv}(Z_t)$ represents an approximate (or complete) linear description of the convex hull of the solutions of the single-period model Z_t.

These new lower bounds LB^{item} and LB^{item}_{time} are never worse, and typically much tighter than the linear relaxation bound LR. The following relations always hold between these bounds.

$$LR \leq LB^{item} \leq LB^{item}_{time} \leq W^{\star} \ .$$

Better lower bounds LB usually allow one to reduce the number of nodes needed to prove optimality, or to obtain good quality solutions. But obtaining these bounds requires more computing time than the time needed to obtain LR because of larger models or more cuts to be added in the cutting plane phase.

For any complex multi-item production planning problem to be solved by an optimization approach, the best reformulation thus depends on

- the existence of reformulation results (approximate or tight compact extended reformulations, valid inequalities, efficient separation algorithms) for the corresponding single-item and/or single-period submodels, and
- the impact of the reformulations on the computing time through a decreased number of branch-and-bound nodes but increased computing time at each node.

The model classification scheme presented next is crucial for an implementation of the decomposition approach. It forces us to present the description, analysis, and structuring of models in a way that facilitates the identification of structured submodels. Then, the systematic reformulation procedure of Section 4.4 identifies the submodels for which reformulation results are available.

Note finally that other optimization methods such as Lagrangian relaxation, Lagrangian decomposition, and Dantzig–Wolfe or column generation, exploit the same decomposition properties of the models. Instead of compact reformulations, these methods require the repeated solution of optimization problems defined over the single-item lot-sizing sets Y^i and the single-period sets Z_t. So to implement these algorithms, it is important to find efficient algorithms to optimize over the single-item/period feasible sets.

The links with these other methods are discussed further in Chapter 6.

4.3 Model Classification

Most practical supply chain planning problems are multi-item, multi-machine, and multi-level, but there exist very few reformulation results concerning such models. Therefore, the main optimization approach in solving such problems has been to integrate existing algorithms and known reformulation results for single-item problems, using a decomposition approach.

We describe here a classification scheme for single-item production planning models that allows one to benefit from this knowledge. Based on this

scheme, the procedure to systematically reformulate and solve production planning models is described in Section 4.4, and illustrated in Section 4.5 on the GW master production scheduling example from Sections 1.2.2 to 1.2.4.

Parts II and III of this book describe the reformulation results according to our scheme for single-item models. Thus for each problem appearing in our classification, we need to describe in detail what results are known and can be used to implement the optimization/decomposition approach for these models. In Part IV we extend our classification to multi-item and multi-level production planning problems, and again present the useful reformulation results that are available. This structured knowledge is then exploited in Part V in solving several industrial cases.

In this section, we describe the basic single-item classification, its notational conventions, and the corresponding mathematical formulations.

4.3.1 Single-Item Classification

Planning problems deal with sizing and timing decisions for purchasing, production, or distribution of lots or batches. An item represents a physical product. The finite planning horizon is divided into time periods, indexed by t, $1 \leq t \leq n$, where n is the given number of time periods.

When considering canonical single-item models, for compactness of notations we use n to represent the length of the planning horizon. This notation is used throughout Sections 4.3 and 4.4, and in Parts II and III of the book. Alternatively, when considering specific production planning instances, we use NT to represent the number of time periods. Similarly, to represent the number of items in the multi-item models studied in Part IV, we use m in canonical models and NI in any particular planning instance.

We start by defining the basic single-item lot-sizing problem (LS). For a single item, we represent by

- d_t the demand to be satisfied in period t, that is forecast demand or customer orders due in period t;
- p'_t the variable or unit production cost in period t;
- h'_t the unit cost for holding one unit in inventory at the end of period t;
- q_t the fixed set-up cost to be paid if there is a positive production in period t;
- C_t the upper bound on production or capacity in period t.

The fixed charge production cost function in period t is characterized by the set-up cost q_t and the unit production cost p'_t.

Problem LS is the problem of finding the production plan for the single item, meeting the demand in every period, and satisfying the capacity restrictions; that is, the production is less than or equal to C_t in every period t, that minimizes the inventory and production costs. Note that in principle a variable amount of initial stock is allowed, at a cost of h'_0 per unit.

Our classification is dictated by the difficulty of solving single-item planning problems, or more precisely by the optimization and reformulation results presented in the literature. There are three fields $PROB$-CAP-VAR. In each field, we use $[x, y, z]^1$ to denote the selection of exactly *one* element from the set $\{x, y, z\}$, and $[x, y, z]^*$ to denote *any* subset of $\{x, y, z\}$. We simply use x, y, z to denote the selection of *all* the elements in the set $\{x, y, z\}$. Fields that are empty are dropped.

4.3.2 Description of the Field *PROB*

In the first field $PROB$, there is a choice of four problem versions $PROB = [LS, WW, DLSI, DLS]^1$.

LS (Lot-Sizing): This is the general problem defined above.

WW (Wagner–Whitin): This is problem LS, except that the variable production and storage costs satisfy $h'_t + p'_t - p'_{t+1} \geq 0$ for $0 \leq t \leq n$, where $p'_0 = p'_{n+1} = 0$. This condition means that, if set-ups occur in both periods t and $t + 1$, then it is more costly to produce in period t and stock till period $t+1$, than to produce directly in period $t+1$. In other words, given the set-ups it always pays to produce as late as possible. This condition is often referred to as the *absence of speculative motive for early production*. We define a new inventory cost as $h_t = h'_t + p'_t - p'_{t+1} \geq 0$ for $0 \leq t \leq n$ (see formulations below).

We name this cost condition WW because it was first introduced in the seminal paper of Wagner–Whitin. It is a little technical, but we show in Part II that it allows one to reduce the running time of the optimization algorithms, and to simplify the reformulation of the planning models. Moreover this condition is very often satisfied by the cost coefficients encountered in practice.

$DLSI$ (Discrete Lot-Sizing with Variable Initial Stock): This is problem LS with the restriction that there is either no production or production at full capacity C_t in each period t.

DLS (Discrete Lot-Sizing): This is problem $DLSI$ without an initial stock variable.

4.3.3 Description of the Field *CAP*

The second field CAP concerns the production limits or capacities $CAP = [C, CC, U]^1$. The three CAP variants of problem $PROB$ are

$PROB$-C (Capacitated): Here the capacities C_t vary over time.

PROB-CC (Constant Capacity): This is the case where $C_t = C$, a constant, for all periods t.

PROB-U (Uncapacitated): This is the case when there is no limit on the amount of the item produced in each period. In the absence of other constraints limiting the total amount produced over all items, this case means that the capacity C_t in each period t suffices to satisfy all the demands up to the end of the horizon.

Before presenting the third field VAR containing the many possible extensions, we present mixed integer programming formulations of the four basic variants with varying capacities *PROB-C*.

4.3.4 Mathematical Formulations for *PROB-CAP*

The standard formulation of LS as a mixed integer program involves the variables

- x_t the amount produced in period t for $1 \le t \le n$,
- s_t the stock at the end of period t for $0 \le t \le n$, and
- $y_t = 1$ if the machine is set up to produce in period t, and $y_t = 0$ otherwise, for $1 \le t \le n$.

We also use the notation $d_{kt} \equiv \sum_{u=k}^{t} d_u$ throughout.

LS-C can be formulated as

$$\min \quad \sum_{t=1}^{n} p'_t x_t + \sum_{t=0}^{n} h'_t s_t + \sum_{t=1}^{n} q_t y_t \tag{4.22}$$

$$s_{t-1} + x_t = d_t + s_t \qquad \text{for } 1 \le t \le n \tag{4.23}$$

$$x_t \le C_t y_t \qquad \text{for } 1 \le t \le n \tag{4.24}$$

$$x \in R^n_+,\ s \in R^{n+1}_+,\ y \in \{0,1\}^n\ , \tag{4.25}$$

and X^{LS-C} denotes the set of feasible solutions to (4.23)–(4.25). Constraint (4.23) represents the flow balance constraint in every period t, the inflows are the initial inventory s_{t-1} and the production x_t, the outflows are the demand d_t and the ending inventory s_t. Constraint (4.24) represents the capacity restriction and also fixes the set-up variable y_t to 1 whenever there is positive production (i.e., $x_t > 0$). This constraint is also called a variable upper bound (VUB) constraint. The objective (4.22) is simply the sum of the set-up, inventory, and variable production costs.

WW-C can be formulated just in the space of the s, y variables as

$$\min \quad \sum_{t=0}^{n} h_t s_t + \sum_{t=1}^{n} q_t y_t \tag{4.26}$$

$$s_{k-1} + \sum_{u=k}^{t} C_u y_u \geq d_{kt} \qquad \text{for } 1 \leq k \leq t \leq n \tag{4.27}$$

$$s \in R_+^{n+1},\ y \in \{0,1\}^n\ , \tag{4.28}$$

and X^{WW-C} denotes the set of feasible solutions to (4.27)–(4.28). To derive this formulation, the constraint (4.23) is used to eliminate x_t from the objective function (4.22). Specifically,

$$\begin{aligned}\sum_{t=1}^{n} p'_t x_t + \sum_{t=0}^{n} h'_t s_t &= \sum_{t=1}^{n} p'_t (s_t - s_{t-1} + d_t) + \sum_{t=0}^{n} h'_t s_t \\ &= \sum_{t=0}^{n} (h'_t + p'_t - p'_{t+1}) s_t + \sum_{t=1}^{n} p'_t d_t \\ &= \sum_{t=0}^{n} h_t s_t + \sum_{t=1}^{n} p'_t d_t\ ,\end{aligned}$$

where $p'_0 = p'_{n+1} = 0$. So, by defining $h_t = h'_t + p'_t - p'_{t+1}$, the objective function (4.22) becomes (4.26) to within the constant $\sum_{t=1}^{n} p'_t d_t$.

Now as $h_t \geq 0$ for all t, it follows that once the set-up periods are fixed (the periods t in which $y_t = 1$), the stocks will be as low as possible compatible with satisfying the demand and respecting the capacity restrictions. Based on this argument it is possible to prove that it suffices to find a minimum cost stock minimal solution in order to solve WW-C, where a *stock minimal* solution is a solution satisfying

$$s_{k-1} = \max(0, \max_{t=k,\ldots,n} [d_{kt} - \sum_{u=k}^{t} C_u y_u]). \tag{4.29}$$

In the proposed formulation (4.26)–(4.28) for WW-C, because of the presence of the initial stock s_0, any combination of set-up periods is feasible, and constraint (4.27) imposes a lower bound on the stock variables. The objective function (4.26), together with $h_t \geq 0$, guarantees that there exists an optimal solution to (4.26)–(4.28) that satisfies (4.29). It follows that the proposed formulation is valid, though its (s, y) feasible region is not the same as that of LS-C. Specifically (s, y) is feasible in (4.27)–(4.28) if and only if there exists (x, s', y) feasible in (4.23)–(4.25) with $s' \leq s$.

Remark 1. Whether the Wagner–Whitin cost condition is satisfied or not, the WW relaxation consisting of the constraints (4.27) is valid for problem LS, and often provides a very good approximation to the convex hull of solutions for problem LS.

Remark 2. Even though each single item subproblem may have WW costs, the existence of other constraints such as multi-item budget (production capacity) constraints or multi-item storage capacity constraints (PQ in the multi-item classification of Section 12.1) destroys the stock minimal solution property for individual items, and thus the items are more correctly classified as LS, rather than WW.

Remark 3. On the other hand, if in a multi-item problem the constraints linking together the different items involve only the set-up or start-up variables (PM in the multi-item classification of Section 12.1), then the stock minimal property of solutions is preserved, and the single items can be classified as WW if their costs satisfy the WW condition.

DLSI-C can be formulated by adding $x_t = C_t y_t$ in the formulation of *LS-C*. By summing constraints (4.23) from 1 up to t, one gets $s_t = s_0 + \sum_{u=1}^{t} x_u - d_{1t}$. Then, after elimination of the variables $s_t \geq 0$ and $x_t = C_t y_t$, we obtain an equivalent formulation of *DLSI-C* just in the space of the s_0 and the y variables, and X^{DLSI_0-C} is used to denote the set of feasible solutions to (4.31)–(4.32),

$$\min \quad h_0 s_0 + \sum_{t=1}^{n} q'_t y_t \tag{4.30}$$

$$s_0 + \sum_{u=1}^{t} C_u y_u \geq d_{1t} \qquad \text{for } 1 \leq t \leq n \tag{4.31}$$

$$s_0 \in R^1_+,\ y \in \{0,1\}^n\ , \tag{4.32}$$

where h_0 and q'_t are the new objective coefficients of variables s_0 and y_t obtained after eliminating the variables s_t and x_t by substitution. Specifically, the objective function (4.22) can be rewritten as

$$\begin{aligned}
&\sum_{t=0}^{n} h'_t s_t + \sum_{t=1}^{n} p'_t x_t + \sum_{t=1}^{n} q_t y_t \\
&\qquad = h'_0 s_0 + \sum_{t=1}^{n} h'_t (s_0 + \sum_{u=1}^{t} C_u y_u - d_{1t}) + \sum_{t=1}^{n} p'_t C_t y_t + \sum_{t=1}^{n} q_t y_t \\
&\qquad = (h'_0 + \sum_{t=1}^{n} h'_t) s_0 - \sum_{t=1}^{n} h'_t d_{1t} + \sum_{t=1}^{n} (q_t + (p'_t + \sum_{u=t}^{n} h'_u) C_t) y_t\ .
\end{aligned}$$

Then defining $h_0 = h'_0 + \sum_{t=1}^{n} h'_t$ and $q'_t = q_t + (p'_t + \sum_{u=t}^{n} h'_u) C_t$, it reduces to (4.30) except for the constant $-\sum_{t=1}^{n} h'_t d_{1t}$.

We also use the notation X^{DLSI_k-C} with $0 \leq k \leq n-1$ to denote the set of solutions of problem $DLSI_k$-C, which is problem *DLSI-C* except that

the initial inventory is located in period k, and production can occur in periods $k+1$ up to n. Problem $DLSI_k$-C involves thus variables $s_k \in R^1_+$ and $y_{k+1}, \ldots, y_n \in \{0,1\}$.

DLS-C can be formulated just in the space of the y variables by fixing $s_0 = 0$:

$$\min \quad \sum_{t=1}^{n} q'_t y_t \tag{4.33}$$

$$\sum_{u=1}^{t} C_u y_u \geq d_{1t} \qquad \text{for all } 1 \leq t \leq n \tag{4.34}$$

$$y \in \{0,1\}^n. \tag{4.35}$$

The set X^{DLS-C} denotes the set of feasible solutions to (4.34)–(4.35). We say that DLS has *Wagner–Whitin costs* if $q'_t \geq q'_{t+1}$ for all t, and without introducing a new problem class we denote this special case as $DLS(WW)$-C.

Observation 4.2 *The constant or uncapacitated problems $PROB$-$[CC, U]^1$ are all polynomially solvable. There is a polynomial dynamic programming algorithm solving LS-CC and the other seven problems can all be seen as special cases.*

All four varying capacity instances $PROB$-C are NP-hard, because all four problems are polynomially reducible to the 0–1 knapsack problem. This means that there are no polynomial algorithms known for them and, from complexity theory, it is very unlikely that there exists a polynomial algorithm for any of them.

We come back to the implications of these observations, to the relationships between these different problems, and to the analysis of algorithms and reformulations for these problems in Part II. So far, we consider that we have different versions of the single-item lot-sizing problem, along with mixed integer programming formulations adapted to each problem class.

4.3.5 Description of the Field VAR

The third field VAR concerns extensions or variants to one of the twelve problems $PROB-CAP$ defined so far; that is, $VAR = [B, SC, ST, LB, SL, SS]^*$. Although such variants can be combined, for simplicity we describe these variants in turn, and give a typical formulation for each problem LS-C-$[B, SC, ST, LB, SL, SS]^1$.

B (Backlogging): Demand must still be satisfied, but it is possible to satisfy a demand later than required. This occurs, for example, when a factory does not have enough capacity to deliver to all customers on time in a given period. Usually, the backlog or shortfall implies a penalty cost proportional to the amount backlogged and to the duration of the backlog.

Note that this backlogging variant is limited to independent or external demand, as the quantity backlogged is only a virtual flow used to model shortfalls in the delivery process and not a physical flow.

SC (Start-Up Costs): It is necessary to accurately model capacity utilization to obtain feasible production plans. This often requires one to model the capacity consumed when a machine starts a production batch, or when a machine switches from one product to another. In these cases, we obtain so-called set-up or start-up time models, changeover time models, or models with sequencing restrictions. However, in many cases, less accurate models involving only set-up or start-up costs are considered. Such models can be seen as obtained by relaxing (in the Lagrangian sense; see Chapter 6) the set-up or start-up time restrictions.

The simplest single-item start-up cost model is the following. If a sequence of set-ups starts in period t, a start-up cost g_t is incurred, which can be seen as the direct start-up cost plus an estimate of the opportunity cost of the start-up time or capacity consumed.

ST (Start-Up Times): As already explained, start-up times are used to model capacity utilization more accurately. The resulting models are more precise, but often more difficult to solve than their start-up cost variant.

If a sequence of set-ups starts in period t, the capacity C_t is reduced by an amount ST_t. We use $ST(C)$ to indicate the start-up time ST is constant over time; that is, $ST_t = ST$ for all t.

LB (Minimum Production Levels): In some problems, in order to guarantee a minimum level of productivity, minimum batch sizes or production levels are imposed. For instance, this feature is often used in combination with start-up costs to approximate start-up time models and avoid small batches in the solutions. This constraint may also be imposed for technological reasons.

If production takes place in period t, a minimum amount LB_t must be produced. We use $LB(C)$ to denote constant lower bounds over time, i.e. $LB_t = LB$ for all t. Note that this leads to variable lower-bound constraints, and not simple lower bounds.

SL (Sales and Lost Sales): In some cases, the demands to be satisfied are not fixed in advance. This occurs, for instance, when capacity is too low to satisfy the total potential demand, or when the selling price does not always cover the marginal cost of production. The optimization problem becomes then a profit maximization problem, with additional sales variables.

In the single-item problem, we model this case in the following way. In addition to the demand d_t that must be satisfied in each period, an additional amount up to u_t can be sold at a unit price of c_t.

Note that this variant can also be used to model the Lost Sales variant in which, as opposed to backlogging, it is possible to not deliver part of the

demand. In this case, the demand from period t that has to be satisfied is d_t, and the additional demand that may be lost or not delivered is u_t. The unit price c_t represents in this case the penalty cost that is avoided for each unit of the additional demand effectively delivered.

SS (Safety Stocks): The last variant that we consider is present in many practical applications, and absent from most scientific publications. When the demand is an output of a forecasting system, it is not known with certainty. Therefore, a minimum amount of planned inventory, called the safety stock, is required at the end of each period so as to handle this uncertainty, and to avoid delivery shortages when actual demand exceeds forecast demand.

The variants described here are common variants included in the field VAR. These plus additional variants concerning either changes in the demand model, production constraints/costs, or sales constraints, are described and analyzed in Chapter 11.

4.3.6 Mathematical Formulations for *PROB-CAP-VAR*

Backlogging

The standard formulation of *PROB-CAP-B* as a mixed integer program involves the additional variables

- r_t the backlog at the end of period t for $t = 1, \ldots, n$.

This cumulated shortfall r_t in satisfaction of the demand in period t is charged at a cost of b'_t per unit. It is assumed throughout that $r_0 = 0$.

This leads to the following formulation for problem *LS-C-B*.

$$\min \sum_{t=0}^{n} h'_t s_t + \sum_{t=1}^{n} b'_t r_t + \sum_{t=1}^{n} p'_t x_t + \sum_{t=1}^{n} q_t y_t \tag{4.36}$$

$$s_{t-1} - r_{t-1} + x_t = d_t + s_t - r_t \qquad \text{for } 1 \le t \le n \tag{4.37}$$

$$x_t \le C_t y_t \qquad \text{for } 1 \le t \le n \tag{4.38}$$

$$x, r \in R^n_+,\ s \in R^{n+1}_+,\ y \in \{0,1\}^n\ , \tag{4.39}$$

and X^{LS-C-B} denotes the set of feasible solutions to the constraints (4.37)–(4.39).

Problem *WW-CAP-B* is problem *LS-CAP-B* except that the costs satisfy the WW cost condition. With backlogging, the costs are said to be *Wagner–Whitin* if both $h_t = p'_t + h'_t - p'_{t+1} \ge 0$ and $b_t = p'_{t+1} + b'_t - p'_t \ge 0$ for $1 \le t \le n-1$. This means that, with respect to backlogging, there are no speculative motives for late production.

As an extension of the simple formulation (4.26)–(4.28) for WW-C, it can be proved that the following formulation involving only the s, r, y variables, is a valid and sufficient formulation for WW-C-B.

$$\min \sum_{t=0}^{n} h_t s_t + \sum_{t=1}^{n} b_t r_t + \sum_{t=1}^{n} q_t y_t \tag{4.40}$$

$$s_{k-1} + r_l + \sum_{u=k}^{l} C_u y_u \geq d_{kl} \qquad \text{for } 1 \leq k \leq l \leq n \tag{4.41}$$

$$s \in \mathbb{R}_+^{n+1},\ r \in \mathbb{R}_+^n,\ y \in \{0,1\}^n\ . \tag{4.42}$$

The notation X^{WW-C-B} is used to represent the set of feasible solutions to the constraints (4.41)–(4.42).

The validity and sufficiency of formulation (4.41)–(4.42) is based on the following nontrivial result. When the objective function (4.40) of WW-C-B satisfies $h_t, b_t \geq 0$ for all t, it suffices to find a minimum cost stock minimal and backlog minimal solution in order to solve WW-C-B, where a solution is called *stock minimal* (resp., *backlog minimal*) if $s_{k-1} = \max_{l \geq k}[d_{kl} - \sum_{u=k}^{l} C_u y_u - r_l]^+$ (resp. if $r_l = \max_{k \leq l}[d_{kl} - \sum_{u=k}^{l} C_u y_u - s_{k-1}]^+$).

After elimination of the $s_1, \ldots, s_n$ variables, $DLSI$-C-B has the following feasible region in the (s_0, r, y) space,

$$s_0 + r_t + \sum_{u=1}^{t} C_u y_u \geq d_{1t} \qquad \text{for } 1 \leq t \leq n \tag{4.43}$$

$$s_0 \in R_+^1, r \in R_+^n, y \in [0,1]^n. \tag{4.44}$$

and X^{DLSI_0-C-B} denotes the set of feasible solutions to (4.43)–(4.44).

Finally, DLS-C-B is obtained from $DLSI$-C-B by setting $s_0 = 0$.

Start-Up Costs

The basic formulation for LS-C-SC requires the introduction of new variables

- $z_t = 1$ if there is a start-up in period t; that is, there is a set-up in period t, but there was not in period $t-1$, and $z_t = 0$ otherwise.

The resulting formulation for LS-C-SC is

$$\min \sum_{t=1}^{n} p_t' x_t + \sum_{t=0}^{n} h_t' s_t + \sum_{t=1}^{n} q_t y_t + \sum_{t=1}^{n} g_t z_t \tag{4.45}$$

$$s_{t-1} + x_t = d_t + s_t \qquad \text{for } 1 \le t \le n \tag{4.46}$$

$$x_t \le C_t y_t \qquad \text{for } 1 \le t \le n \tag{4.47}$$

$$z_t \ge y_t - y_{t-1} \qquad \text{for } 1 \le t \le n \tag{4.48}$$

$$z_t \le y_t \qquad \text{for } 1 \le t \le n \tag{4.49}$$

$$z_t \le 1 - y_{t-1} \qquad \text{for } 1 \le t \le n \tag{4.50}$$

$$x \in R_+^n,\ s \in R_+^{n+1},\ y, z \in \{0,1\}^n\ , \tag{4.51}$$

and the set of feasible solutions to (4.46)–(4.51) is denoted by $X^{LS-C-SC}$. We assume that y_0, the state of the machine at time 0, is given as data. The additional constraints (4.48)–(4.50) define the values of the additional start-up variables. These constraints are a linearization of the constraint $z_t = y_t(1 - y_{t-1})$, for all t. There can be a start-up in period t (i.e., $z_t = 1$) only if there is a start-up in period t (see (4.49)) and no start-up in period $t-1$ (see (4.50)), and there must be a start-up in period t if both events occur simultaneously (see (4.48)).

The formulations of $[WW, DLSI, DLS]^1$-C-SC, as well as their corresponding feasible sets $X^{[WW,DLSI_0,DLS]^1-C-SC}$, are obtained by just adding the constraints (4.48)-(4.50) and the integrality restrictions $z \in \{0,1\}^n$ to the formulations $[WW,DLSI,DLS]^1$-C given above.

Start-Up Times

The basic formulation for *LS-C-ST* requires the same start-up variables z_t as the start-up cost model *LS-C-SC*. The formulation for *LS-C-ST* is the same as for *LS-C-SC* ((4.45)-(4.51)), except that the variable upper bound constraint (4.47) has to be replaced by the constraint

$$x_t \le C_t y_t - ST_t z_t \quad \text{for } 1 \le t \le n.$$

Minimum Production Levels

The basic formulation for *LS-C-LB* requires no additional variables. The formulation for *LS-C-LB* is the same as for *LS-C* ((4.22)–(4.25)), augmented with the variable lower bound constraint

$$x_t \ge LB_t y_t \quad \text{for } 1 \le t \le n.$$

Sales

The standard formulation of *LS-C-SL* as a mixed integer program involves the additional variables

- v_t the amount sold in period t, on top of the fixed demand d_t, for $1 \leq t \leq n$,

and is given by

$$\max \sum_{t=1}^{n}(c_t v_t - p_t x_t) \; - \; \sum_{t=0}^{n} h'_t s_t - \sum_{t=1}^{n} q_t y_t \tag{4.52}$$

$$s_{t-1} + x_t = d_t + v_t + s_t \qquad \text{for } 1 \leq t \leq n \tag{4.53}$$

$$x_t \leq C_t y_t \qquad \text{for } 1 \leq t \leq n \tag{4.54}$$

$$v_t \leq u_t \qquad \text{for } 1 \leq t \leq n \tag{4.55}$$

$$x, v \in R^n_+, s \in R^{n+1}_+, y \in \{0,1\}^n \; , \tag{4.56}$$

where the objective (4.52) maximizes the contribution to profit, and the flow balance constraint (4.53) is updated to take the sales outflow into account. Constraint (4.55) models the simple upper bound on sales.

Safety Stocks

To incorporate this requirement, we just need to add a simple lower bound SS_t on the stock level at the end of period t; that is, $s_t \geq SS_t$ for all periods t with $1 \leq t \leq n$

4.3.7 The Classification *PROB-CAP-VAR*

We have described the three fields $PROB$-CAP-VAR of the single-item lot-sizing classification, namely,

$$[LS, WW, DLSI, DLS]^1 - [C, CC, U]^1 - \\ [B, SC, ST, ST(C), LB, LB(C), SL, SS]^*$$

where one entry is required from each of the first two fields, and any number of entries from the third.

For instance, WW-U (in place of WW-U-$\emptyset$) denotes the uncapacitated Wagner–Whitin problem, whereas $DLSI$-CC-B, ST denotes the constant capacity discrete lot-sizing problem with initial stock variable, backlogging, and start-up times.

Observation 4.3 *It turns out that almost all the variants $PROB$-$[CC, U]^1$-VAR are still polynomially solvable if the start-up times or lower bounds, if any, are constant (versions $ST(C), LB(C)$).*

This terminates the description of the classification for single-item problems. It is clearly beyond the scope of this description to give a complete mathematical programming formulation of all possible variants from the classification. These different formulations are described in more detail in Parts II and III.

4.4 Reformulation Results: What and Where

In this section, we list first the reformulation results available (the "What") for the most common or standard single-item lot-sizing problems, classified according to the scheme described in Section 4.3.

More precisely, we give the results in the form of three reformulation tables for the uncapacitated and constant capacity single-item models:

- The basic models $[LS, WW, DLSI, DLS]^1$-$[U, CC]^1$ without variants.
- The models with backlogging $[LS, WW, DLSI, DLS]^1$-$[U, CC]^1$-B.
- The models with start-up costs $[LS, WW, DLS]^1$-$[U, CC]^1$-SC.

Note that we do not give reformulation tables for models with varying capacity (value C of the field CAP) because there are no complete reformulation results available for these high-level relaxations, due to the fact all these models define NP-hard optimization problems.

For variants other than backlogging or start-up costs, there are only a few results available. The partial reformulation results known for these models, and the reformulation results for lower-level relaxations contained in these models, are given in Parts II to IV of the book.

For each model in these tables, we indicate the reformulation results in three sections.

- **Formulation** reports on the existence and the size (order of the number of constraints and variables) of tight and compact linear a priori reformulations.
- **Separation** gives the complexity of the separation algorithms for the tight reformulations in the original variable space.
- **Optimization** contains the complexity of the best optimization algorithm known for the model.

In each case, we indicate a reference to the research paper or publication containing, to our knowledge, the original result, as well as a pointer to the section in this book where the result is described in detail.

The tables also indicate the missing results. An *asterisk* * indicates that the family of inequalities only gives a partial description of the convex hull of solutions. A *triple asterisk* *** indicates that we do not know of any result specific to the particular problem class.

Even if they are not used in a direct solution approach by branch-and-bound/cut, we have included results for the associated optimization problems because they are very much related to the other results, and because other optimization methods such as Lagrangian relaxation or Dantzig–Wolfe decomposition require the solution of the optimization version of these standard models.

Finally, we conclude this section by providing a *reformulation procedure* (the "Where") indicating how to use the results in the tables, and build improved formulations for complex production planning models. Note that this

procedure requires the use of the classification scheme and reformulation tables, but does not require any knowledge about the mathematical description or analysis of the reformulations.

4.4.1 Results for $PROB$-$[U, CC]$

In Table 4.4 we present results for the models $[LS, WW, DLSI, DLS]$-$[U, CC]$. Note that the entries $[DLSI, DLS]$-U have been left blank as the results and algorithms are trivial. In the **Formulation** entries for LS-U, FL denotes a facility location reformulation, SP denotes a shortest path reformulation, and MC denotes the multi-commodity reformulation already presented in Section 4.1.1.

Table 4.4. Models $PROB$-$[U, CC]$

	LS	WW	$DLSI$	DLS
Formulation	$Cons \times Vars$	$Cons \times Vars$	$Cons \times Vars$	$Cons \times Vars$
U	$SP\ O(n) \times O(n^2)$ $FL\ O(n^2) \times O(n^2)$ $MC\ O(n^2) \times O(n^2)$ *Section* 7.4.2 [100, 61, 145]	$O(n^2) \times O(n)$ *Section* 7.5 [140]	$--$	$--$
CC	$O(n^3) \times O(n^3)$ *Section* 9.6.3 [176]	$O(n^2) \times O(n^2)$ *Section* 9.5.3 [140]	$O(n) \times O(n)$ *Section* 9.4.2 [125, 140]	$O(n) \times O(n)$ *Section* 9.3.1 *Folklore*
Separation				
U	$O(n \log n)$ *Section* 7.4.1 [23]	$O(n)$ *Section* 7.5 [140]	$--$	$--$
CC	$*$ *Section* 9.6.2/3 [139]	$O(n^2 \log n)$ *Section* 9.5.2 [140]	$O(n \log n)$ *Section* 9.4.1 [85, 125, 140]	$O(n)$ *Section* 9.3.1 *Folklore*
Optimization				
U	$O(n \log n)$ *Section* 7.3 [3, 63, 187]	$O(n)$ *Section* 7.3 [3, 63, 187]	$--$	$--$
CC	$O(n^3)$ *Section* 9.6.1 [71, 171]	$O(n^2 \log n)$ *Section* 9.5.1 [178]	$O(n^2 \log n)$ *Section* 9.4 [178]	$O(n \log n)$ *Section* 9.3.2 [178]

Reading these tables is straightforward. Looking at the WW–CC entry in the **Formulation** block, we see that, for the problem with Wagner–Whitin costs and constant capacities, there is an extended formulation with $O(n^2)$ constraints and $O(n^2)$ variables that gives the convex hull. Details are to be

found in Section 9.5.3. We see also in the WW–CC entry in the **Separation** block that there is a separation algorithm for the same problem whose running time is $O(n^2 \log n)$. Finally we see from the WW–CC entry in the **Optimization** block that the fastest known algorithm to find an optimal solution for this problem runs in $O(n^2 \log n)$.

4.4.2 Results for Backlogging Models *PROB*-[*U*, *CC*]-*B*

Now we consider the same problems but with backlogging. The results are given in Table 4.5.

Table 4.5. Models with Backlogging $PROB$-$[U, CC]$-B

	LS	WW	$DLSI$	DLS
Formulation	$Cons \times Vars$	$Cons \times Vars$	$Cons \times Vars$	$Cons \times Vars$
U	$SP\ O(n) \times O(n^2)$ $FL\ O(n^2) \times O(n^2)$ *Section* 10.2.2 [22, 137]	$O(n^2) \times O(n)$ *Section* 10.2.3 [140]	$--$	$--$
CC	$O(n^3) \times O(n^3)$ *Section* 10.3.4 [178, 180]	$O(n^3) \times O(n^2)$ *Section* 10.3.3 [125, 178]	$O(n^2) \times O(n)$ *Section* 10.3.2 [125, 176]	$O(n) \times O(n)$ *Section* 10.3.1 [125]
Separation				
U	$*$ *Section* 10.2.2 [137]	$O(n^3)$ *Section* 10.2.3 [140]	$--$	$--$
CC	$*$ *Section* 10.3.4	$*$ *Section* 10.3.3 [134, 104, 125]	$O(n^3)$ *Section* 10.3.2 [125]	$O(n)$ *Section* 10.3.1 [125]
Optimization				
U	$O(n \log n)$ *Section* 10.2.1 [3, 63, 187]	$O(n)$ *Section* 10.2.3 [3, 63, 187]	$--$	$--$
CC	$O(n^3)$ *Section* 10.3.4 [176]	$O(n^3)$ *Section* 10.3.3 [176]	$O(n^2 \log n)$ *Section* 10.3.2 [176]	$O(n^2)$ *Section* 10.3.1 [176]

4.4.3 Results for Start-Up Cost Models *PROB*-[*U*, *CC*]-*SC*

Finally we list in Table 4.6 the results for problems with start-up costs.

$DLS(WW)$ refers to the special case of DLS-CC-SC with just set-up and start-up costs in which the set-up costs are non-increasing over time (i.e., $q_t \geq q_{t+1}$; see Section 10.5.1).

Table 4.6. Models with Start-Up Costs $PROB$-$[U, CC]$-SC

	LS	WW	DLS
Formulation	$Cons \times Vars$	$Cons \times Vars$	$Cons \times Vars$
U	$SP(SC)$ $O(n^2) \times O(n^2)$ $FL(SC)$ $O(n^3) \times O(n^2)$ *Section* 10.4.2 [170, 191]	$O(n^2) \times O(n)$ *Section* 10.4.3 [140]	$--$
CC	$***$	$***$	$O(n^2) \times O(n^2)$ (WW) $O(n^2) \times O(n)$ *Section* 10.5.1 [165, 163]
Separation			
U	$O(n^3)$ *Section* 10.4.2 [170, 191]	*Exercise*10.13 *Section* 10.4.3 [140]	$--$
CC	$O(n^2)$ $*$ *Section* 10.5 [46]	$***$	$*$ *Section* 10.5.1 [164]
Optimization			
U	$O(n \log n)$ *Section* 10.4.1 [3, 63, 187]	$O(n)$ [3, 63, 187]	$--$
CC	$O(n^4)$ *Section* 10.5 [71]	$***$	$O(n^2)$ (WW) $O(n \log n)$ *Section* 10.6 [67, 147, 164]

Finally there is a reformulation for WW-U-B, SC, described in Section 10.6, with $O(n^2)$ constraints and $O(n)$ variables.

4.4.4 The Reformulation Procedure

Here we present general guidelines on how to use the classification scheme and the reformulation tables in order to obtain good or state-of-the-art formulations for production planning models.

We demonstrate the approach in detail in the next section on the Master Production Scheduling Example from Chapter 1, and on elementary case studies in Chapter 5.

Rule 1. Construct an initial model and a mathematical formulation using the classification scheme from Section 4.3. In particular, characterize or classify the single-item models as $PROB$-CAP-VAR.

Rule 2. For each single-item model, select appropriate reformulations by identifying the closest cell or cells in the reformulation tables.
The choice of a reformulation depends often on a compromise between its quality or tightness and its size. Therefore, several reformulations can be selected. From a given cell identified from the classification, we can move to other cells in order to obtain valid or allowed reformulations of the model. The allowed moves are

- *move upwards* $CC \Rightarrow U$, usually performed to reduce the size of the reformulation or the number of cuts generated.
- *towards the right* $LS \Rightarrow WW$, usually to reduce the size of the reformulation or the number of cuts generated.
- *towards the right* $WW \Rightarrow \{DLSI_k\}_{k=0,\ldots,n-1}$.
- *towards the left* $WW \Rightarrow LS$.

Rule 3. The different reformulations identified should then be implemented, tested, and compared in terms of solution quality and computing time.

The allowed moves from cell to cell given in Rule 2, as well as some other moves, are justified by the following relations that exist between the sets of feasible solutions associated with the problems in the classification; see Section 4.3.

$$X^{prob-cap-SC} \subseteq X^{prob-cap} ,$$

$$X^{prob-CC-var} \subseteq X^{prob-U-var} ,$$

$$X^{LS-cap-var} \subseteq X^{WW-cap-var} \subseteq \bigcap_{k=0}^{n-1} X^{DLSI_k-cap-var} ,$$

where in each relation *prob*, *cap*, and *var* represent any fixed value of the fields $PROB$, CAP, and VAR, respectively. For instance, as any solution of $X^{prob-CC-var}$ is included in the larger set $X^{prob-U-var}$, any valid constraint or formulation for the larger set $X^{prob-U-var}$ is also valid for $X^{prob-CC-var}$, and thus the move $CC \Rightarrow U$ is allowed.

The move $WW \Rightarrow LS$ is justified by the discussion and remarks in Section 4.3 relative to the choice between classification LS or WW for the field $PROB$. In a multi-item lot-sizing problem where the single items satisfy the WW cost condition, the classification and formulation LS are more appropriate when additional constraints (such as linking capacity constraints) destroy the stock minimal characteristic of optimal solutions.

As an illustration, consider a multi-item single-level single-machine problem. Suppose that the subproblem for each item is classified as WW-CC-B.

- We identify first the cell WW-CC-B in Table 4.5. A reformulation is proposed, but $O(n^3)\times O(n^2)$ appears very large, because this reformulation must be applied individually to all items.

- We can move upwards from CC to U in Table 4.5 to find a relaxation. The relaxation WW-U-B is obtained for which a tight and more compact $O(n^2) \times O(n)$ reformulation is indicated.
- We can move towards the right in Table 4.5 to find another relaxation. We obtain the relaxations $DLSI_k$-CC-B, for $k = 0, \ldots, n-1$, for which a tight $O(n^2) \times O(n)$ reformulation is again known for each k. However, this leads to an $O(n^3) \times O(n^2)$ formulation, which is again large.

4.5 A Production Planning Example: Reformulation and Solution

We have already illustrated on a MPS example in Section 1.2.4 that the structure of a MIP formulation can be used in order to improve both the quality of the solution and the final duality gap (see Table 1.5). Such improvements were based on the reformulation of simple (low-level) structures embedded in the problem, such as single mixed integer constraints or single-node flow structures (see Chapter 8). Moreover, they are obtained automatically by using state-of-the-art branch-and-cut solvers.

Here we show how to profit from the classification scheme to recognize more global structures that are specific to production planning problems. It is then possible to use the known reformulation results for these canonical planning structures in order to obtain an even better formulation of the initial problem.

As a simple and basic illustration of this principle (more comes later in the case studies in Chapter 5 and in Part V), we analyze the initial formulation (1.1)–(1.7) of our MPS example and observe that the Wagner–Whitin cost condition is satisfied because there are no production costs and there are positive inventory costs. Moreover, constraints (1.2)–(1.4) define an uncapacitated lot-sizing structure for each product and constraint (1.3) defines safety stocks for each item. Therefore each single-item submodel is classified as

$$WW{-}U{-}SS.$$

Observe that the single-item problems could be classified as LS-U-SS because the capacity constraints linking the different items are likely to destroy the stock minimal structure of optimal solutions (see the discussion and remarks in Section 4.3 relative to the choice between classification LS or WW for multi-item problems).

We illustrate here how to use some known a priori reformulation results for these single-item submodels. These reformulations are given here for completeness, but they are analyzed in depth in Parts II and III.

Removing the Safety Stocks

First, note that the reformulation Table 4.4 does not include the safety stock variant. So, before applying the $WW{-}U$ reformulation with $O(n^2)$ constraints

and $O(n)$ variables from Table 4.4, we apply a standard linear programming trick to remove the simple lower bound on the inventory variables, that is, to remove the safety stocks.

The inequality $s_t^i \geq SS_{t-1}^i - D_t^i$ always holds because the entering stock of item i in period t that is not used to satisfy some demand in period t must be part of the inventory at the end of period t. Therefore, and without loss of generality, we can tighten the safety stock for each item i and for $t = 1, \cdots, NT$ by setting

$$SS_t^i \ := \ \max\{SS_{t-1}^i - D_t^i, \ SS_t^i\},$$

where SS_0^i is the initial inventory of item i.

Then, we can eliminate the lower bounds on inventory by defining net inventory variables $ns_t^i := s_t^i - SS_t^i \geq 0$ for all i and t. After replacing the inventory variables by the net inventory variables (i.e., replacing s_t^i everywhere by $ns_t^i + SS_t^i$), we obtain the following equivalent formulation.

$$\min \ \sum_i \sum_t \ ns_t^i \ + \ \sum_i \sum_t \ SS_t^i \tag{4.57}$$

$$ns_{t-1}^i + x_t^i = ND_t^i + ns_t^i \qquad \text{for all } i,t \tag{4.58}$$

$$x_t^i \leq M_t^i y_t^i \qquad \text{for all } i,t \tag{4.59}$$

$$\sum_i \alpha^{i1} x_t^i + \sum_i \beta^i y_t^i \leq C^1 \qquad \text{for all } t \tag{4.60}$$

$$\sum_{i \in F^k} \alpha^{ik} x_t^i \leq C^k \qquad \text{for all } t \ \text{ and } k = 2,3 \tag{4.61}$$

$$ns_0^i = 0 \ , \ ns_t^i \in \mathbb{R}_+^1 \qquad \text{for all } i,t \tag{4.62}$$

$$x_t^i \in \mathbb{R}_+^1, \ y_t^i \in \{0,1\} \qquad \text{for all } i,t, \tag{4.63}$$

where $ND_t^i \ := \ D_t^i + SS_t^i - SS_{t-1}^i \geq 0$ is the net demand of item i in period t, and where the upper bound M_t^i on the production of item $i \in F^k$ in period t in constraint (4.59) is taken as

$$M_t^i = \min\{\sum_{l=t}^{NT} ND_l^i \, , \ \frac{C^1 - \beta^i}{\alpha^{i1}} \, , \ \frac{C^k}{\alpha^{ik}} \}.$$

Extended Reformulation *WW-U*

Each single-item model (4.57)–(4.59) and (4.62)–(4.63) in the above formulation is classified as *WW-U*. Table 4.4 indicates the existence of the following linear reformulation with $O(n^2)$ constraints and $O(n)$ variables for this *WW-U* model (written for item i, translated directly for the net demand data ND_t^i and the net inventory variables ns_t^i); see Chapter 7.

$$ns^i_{t-1} + x^i_t = ND^i_t + ns^i_t \qquad \text{for all } t \tag{4.64}$$

$$ns^i_{t-1} \geq \sum_{j=t}^{l} ND^i_j (1 - \sum_{u=t}^{j} y_u) \qquad \text{for all } t, l \tag{4.65}$$

$$ns^i_t, x^i_t \in \mathbb{R}^1_+,\ y^i_t \in [0,1] \qquad \text{for all } i, t \tag{4.66}$$

The $O(n^2)$ constraints (4.65) impose that the stock at the end of period $t-1$ must contain the demand of period $j \geq t$ if there are no set-ups in periods t up to j (i.e., if $\sum_{u=t}^{j} y_u = 0$).

The first reformulation consists of constraints (4.57)–(4.63), plus the constraints (4.65) for all items instead of the constraints (4.59). It is easily implemented in Mosel. The results obtained with the Xpress-MP Optimizer using this a priori reformulation are compared in Table 4.7 with the results obtained using the initial or basic formulation (4.57)–(4.63), with and without the Xpress-MP system cuts. Column "LP Val." gives the initial linear relaxation or lower-bound value before the Xpress-MP cuts, and column "XLP Val." gives the lower bound obtained at the root node after the addition of of Xpress-MP cuts.

Table 4.7. Extended Reformulation *WW-U* for the GW MPS Example

Algorithm Formulation	Vars Cons	LP Val.	XLP Val. Ncuts	Best LB Best UB	Best UB t. (secs) Gap (%)
Basic form. B & B	540	2893	2893	3341	0
(w/o Xpress-MP cuts)	405		0	6415	47.92
Basic form. B & C	540	2893	5481	5614	56
(with Xpress-MP cuts)	405		239	5746	2.30
WW – U B & C	540	5395	5496	5652	269
(with Xpress-MP cuts)	1845		18	5732	1.40

$NI = 12$ and $NT = 15$. Maximum 600 second runs.

The optimization was stopped after 600 seconds. With the *WW-U* reformulation, we obtain a slightly better feasible solution (see column "Best UB"), and better initial (see column "XLP Val.") and final lower bounds (see column "Best LB"). The column "Best UB t." gives the time in seconds to find the best feasible solution. The duality gap is reduced to 1.40%. Note that these results are obtained with the combination of the generic Xpress-MP cuts (with default branch-and-cut parameter settings) and the specific production planning reformulations.

Other Extended Reformulations

As we already observed, the single-item problems can also be classified as *LS-U* because the capacity constraints linking the different items are likely to

destroy the stock minimal structure of optimal solutions. Therefore the known reformulations for model *LS-U* given in Table 4.4 (namely the facility location (FL), shortest path (SP), and multi-commodity (MC) reformulations; see Chapter 7) could also be used and tested (Rule 3 of the reformulation procedure).

We have described the multi-commodity reformulation at the beginning of this chapter. As another example, the facility location reformulation for the single-item *LS-U* model (4.58)–(4.59), (4.62)–(4.63) (without lower bounds on the net inventory) is defined on the extended variable space x^i_{tl}, for all items i, periods t and $l \geq t$, where x^i_{tl} represents the amount of item i produced in period t to satisfy net demand in period $l \geq t$.

Using the facility location reformulation, and the substitutions $x^i_t = \sum_{l \geq t} x^i_{tl}$ and $ns^i_t = \sum_{k=1}^{t} \sum_{l=t+1}^{NT} x^i_{kl}$, the final facility location reformulation of (4.57)–(4.63) is

$$\min \quad \sum_i \sum_t \sum_{l \geq t} (l-t) x^i_{tl} \; + \; \sum_i \sum_t \; SS^i_t \tag{4.67}$$

$$\sum_{t=1}^{l} x^i_{tl} = ND^i_l \qquad \text{for all } i, l \tag{4.68}$$

$$x^i_{tl} \leq ND^i_l y^i_t \qquad \text{for all } i, t, l \text{ with } l \geq t \tag{4.69}$$

$$\sum_i \sum_{l \geq t} \alpha^{i1} x^i_{tl} + \sum_i \beta^i y^i_t \leq C^1 \qquad \text{for all } t \tag{4.70}$$

$$\sum_{i \in F^k} \sum_{l \geq t} \alpha^{ik} x^i_{tl} \leq C^k \qquad \text{for all } t \text{ and } k = 2, 3 \tag{4.71}$$

$$x^i_{tl} \in \mathbb{R}^1_+, \; y^i_t \in \{0, 1\} \qquad \text{for all } i, t, l \text{ with } l \geq t \; . \tag{4.72}$$

The shortest path reformulation is derived directly from the dynamic programming algorithm used to solve *LS-U*, and is described in Chapter 7.

As a last reformulation, we can also implement and test the $O(n^2) \times O(n^2)$ extended reformulation for the single-item constant capacity model *WW-CC* referred to in Table 4.4 and described in Chapter 9. For some items, the total demand over the planning horizon is larger than the production capacity of one period. Therefore, for each item i, with $i \in F^k$, one can define a constant upper bound on production

$$U^i = \min\{\sum_{t=1}^{NT} ND^i_t, \; \frac{C^1 - \beta^i}{\alpha^{i1}}, \; \frac{C^k}{\alpha^{ik}} \; \},$$

such that $x^i_t \leq U^i y^i_t$ is valid for all t. In any case, model *WW-CC* is larger than, but at least as strong, as model *WW-U*.

The results obtained using these extended reformulations with the Xpress-MP Optimizer are compared in Table 4.8 with the results obtained using the

initial or basic formulation. All the results have been obtained with the default branch-and-cut system from Xpress-MP.

Table 4.8. Extended Reformulations for the GW MPS Example

Algorithm Formulation	Vars Cons	LP Val.	XLP Val. Ncuts	Best LB Best UB	Best UB t. (secs) Gap (%)
Basic form. B & B	540	2893	2893	3341	0
(w/o Xpress-MP cuts)	405		0	6415	47.92
Basic form. B & C	540	2893	5481	5614	56
(with Xpress-MP cuts)	405		239	5746	2.30
WW-U B & C	540	5395	5496	5652	269
(with Xpress-MP cuts)	1845		18	5732	1.40
LS-U (MC) B & C	2880	5395	5503	5667	88
(with Xpress-MP cuts)	2925		26	5732	1.13
LS-U (FL) B & C	1620	5395	5526	5702	534
(with Xpress-MP cuts)	1665		59	5730	0.49
LS-U (SP) B & C	1620	5395	5486	5672	419
(with Xpress-MP cuts)	417		22	5730	1.01
WW-CC B & C	2160	5395	5480	5651	319
(with Xpress-MP cuts)	2205		23	5732	1.41

$NI = 12$ and $NT = 15$. Maximum 600 second runs.

We observe in Table 4.8 that the results obtained with the different reformulations are similar. In 600 seconds, the best lower bound is achieved by the facility location reformulation, and the best feasible solution is obtained by the shortest path and the facility location reformulations. As expected, the *LS-U* reformulations tend to lead to (slightly) better lower bounds than the *WW-U* reformulation. The capacitated model *WW-CC* has no additional effect, probably because the capacity is always shared between items and the bound U^i on the individual production batches is not binding. The duality gap computed with the best lower and upper bounds among all the reformulations is 0.49%.

Given the good results obtained with the facility location reformulation, we solved the problem with this reformulation without any time limit, in order to obtain the optimal solution. The optimal solution is the solution of value 5730 found in less than 600 seconds, and it took 1195 seconds and 386,700 nodes in total to prove its optimality.

Reformulations in the Original Variable Space by Cutting Planes

We can observe in Table 4.8 that the better results (lower and upper bounds) have been obtained at the price of a large increase in the size of the formulation. This may slow down the solution of the linear relaxations, and reduce

the number of branch-and-bound nodes evaluated within the time limit of 600 seconds.

An alternative leading to the same lower bound at the root node would be to reformulate the single-item models *LS-U* using the complete linear reformulation by valid inequalities in the original variable space (4.14)–(4.18) described in Section 4.1.2. It involves an exponential number of (l, S) constraints (4.17) that can be added using the separation algorithm described in Section 4.1.2.

We have tested this approach at the root node, starting from the basic formulation (4.57)–(4.63) where the safety stocks have been removed, by performing the following:

- Solving the linear relaxation;
- Solving the separation problem for each item i and each period l;
- Adding to the formulation each violated (l, S) inequality identified;
- Re-optimizing the new linear relaxation (only after the generation of cuts for all items i and all periods l);
- Solving again the separation problem for each item and period;
- Repeating this procedure until no more violated (l, S) inequalities are generated.

This can be easily implemented in the Mosel modeling language. On our MPS test problem, this procedure requires 14 passes (i.e., 14 iterations of cut generation for all items and periods with a single reoptimization) and generates 933 violated (l, S) cuts in total, in about 20 seconds. In order to reduce the size of the model, these cuts have been added as model cuts; that is, inactive cuts are removed from the model and put into a cut pool. In this way, only 458 of the cuts are kept in the final formulation at the top node.

Then the resulting formulation at the root node is passed to Xpress-MP, and the default MIP solver is used. The results of this cut-and-branch approach are given in Table 4.9.

Table 4.9. Cutting Plane Reformulation for the GW MPS Example

Algorithm Formulation	Vars Cons	LP Val.	CPLP Val. Ncuts	XLP Val. Ncuts	Best LB Best UB	Best UB t. Gap (%)
Basic form. B & C	540	2893	5395	5479	5672	492
with (l, S) cuts and	405		458	52	5730	1.01
Xpress-MP cuts						

$NI = 12$ and $NT = 15$. Maximum 600 second runs.

We observe in Table 4.9 that the lower bound obtained with the 458 (l, S) inequalities generated as cuts at the root node before the addition of Xpress-MP cuts (see column "CPLP Val.") is effectively the same as the lower bound obtained with the extended reformulations (column "LP Val." in Table 4.8).

This holds because all reformulations define complete linear descriptions of the single item models.

Although this formulation is of the same quality as and of smaller size than the extended formulations, which allows one to evaluate more nodes in the same amount of time, the best lower bound obtained after 600 seconds is not better than with the extended reformulations. This may be due to the fact that we do not generate additional violated (l, S) inequalities in the branch-and-bound tree, and therefore the bounds in the tree may be worse than with the tight extended reformulations.

Note also that the optimal feasible solution is again found in less than 600 seconds.

Heuristic Primal Solutions

The reformulations used and tested so far are mainly aimed at improving the lower or dual bound on the objective function, but are not specifically designed to produce good feasible or primal solutions quickly.

So to obtain better upper bounds, we apply the relax-and-fix construction heuristic and the relaxation-induced neighborhood search improvement heuristic described in Section 3.6.

For relax-and-fix we have decomposed the planning horizon into three equal parts.

- In the first iteration, we relax the set-up variables for periods in $\{6, \ldots, 15\}$, solve the resulting MIP^1, and then fix the set-up decisions for periods in $\{1, \ldots, 5\}$.
- In the second iteration, with the fixed set-up decisions for periods in $\{1, \ldots, 5\}$, we relax the set-up variables for periods in $\{11, \ldots, 15\}$, solve the resulting MIP^2 and we additionally fix the set-up decisions for periods in $\{6, \ldots, 10\}$.
- In the third and last iteration, with the fixed set-up decisions for periods in $\{1, \ldots, 10\}$, we optimize the set-up decisions for periods in $\{11, \ldots, 15\}$.

This corresponds to $R = 3$, $Q^1 = \{1, \ldots, 5\}$, $Q^2 = \{6, \ldots, 10\}$, $Q^3 = \{11, \ldots, 15\}$, $U^1 = U^2 = \emptyset$ in the notation of Section 3.6.2.

To test the ability of the algorithm to generate good solutions quickly, we have limited the computation time of each iteration to maximum 40 seconds. So, we fix variables at their values in the best solution obtained after maximum 40 seconds, and the relax-and-fix algorithm takes maximum 120 seconds in total. Note that the only true lower bound produced by this relax-and-fix procedure is the best lower bound obtained at the end of the first iteration (solution of MIP^1) before any variable fixing.

We have implemented the relax-and-fix procedure in Mosel. This simply requires three successive runs of almost identical models. The only modifications are the status of the binary variables from relaxed to binary, and from

binary to fixed. The results obtained are given in Table 4.10 using the WW-U and WW-CC reformulations.

First, the running times of the relax-and-fix heuristic are only 41 and 43 seconds, respectively, with formulations WW-U and WW-CC, because the time limit of 40 seconds is reached only for the second iteration MIP^2. Next, the relax-and-fix heuristic produces feasible solutions quickly, but of relatively moderate quality ("R&F Val.") compared to those obtained in 600 seconds without this procedure (see "Best UB" in Table 4.9). Also, the lower bounds obtained are very weak (see "Best LB" in Tables 4.10 and 4.9).

Table 4.10. Heuristic solution for the GW MPS Example

Formulation Algorithm	Vars Cons	LP Val.	Best LB	R&F Val. RINS Val.	R&F Time RINS Time
WW-U B&C/R&F/RINS	540	5395	5429	5928	41
(with Xpress-MP cuts)	1845			5743	2
WW-CC B&C/R&F/RINS	2160	5395	5429	5770	43
(with Xpress-MP cuts)	2205			5730	2

$NI = 12$ and $NT = 15$; Maximum 160 second runs.

We have also tested the relax-and-fix heuristic on the basic formulation (4.57)–(4.63). It failed to produce a feasible solution because the program obtained at iteration 2, after fixing the set-up decisions for periods $\{1, \ldots, 5\}$, was infeasible. Due to the weak relaxed model for periods $\{6, \ldots, 15\}$ (i.e., no reformulation is used), the set-up decisions obtained for the first periods do not anticipate the capacity problems in later periods and lead to an infeasible solution.

Therefore, it appears to be very important for the feasibility and quality of the relax-and-fix procedure to start with a good formulation of the problem, that is, with a good linear relaxation, or to find another way to anticipate the capacity restrictions in later periods.

Finally we have implemented and tested the relaxation-induced neighborhood search improvement heuristic described in Section 3.6.2. Specifically, we fix the set-up variables that have the same value (0 or 1) in the linear relaxation (root node solution) and in the solution obtained by relax-and-fix. We then solve the resulting MIP using the default Xpress-MP, with a time limit of 40 seconds (maximum 160 seconds, including relax-and-fix). The results in Table 4.10 show that the RINS procedure is able to improve the relax-and-fix solution, and even once to produce the optimal solution ("RINS Val."), in almost no additional running time.

The next chapter shows how to use the classification scheme and the reformulation procedure in practice, and includes two small case studies.

The objective of Parts II to IV is to present all the available reformulation approaches and results in a systematic way. Then, as in our illustrative example, Part V uses these results with the support of the classification scheme to solve industrial case studies.

Exercises

Applications and exercises relative to the classification scheme and the reformulation procedure are given in the case studies of Chapters 5 and 14.

Notes

Sections 4.1 The multi-commodity reformulation for fixed charge network flow problems, implemented and tested in Section 4.1.1, was proposed by Rardin and Choe [145].

Sections 4.3 and 4.4 The classification scheme and the reformulation tables are taken from Wolsey [194]. An earlier and somewhat different classification scheme has been proposed by Bitran and Yanasse [28], and these authors also prove that the four varying capacity problems $PROB\text{-}C$ are NP-hard, because these problems are polynomially reducible to the 0–1 knapsack problem.

Section 4.5 The formulations and results presented here (and in Section 4.1) have been implemented and obtained using the Mosel algebraic modeling language (version 1.4.1) and the default version of the Xpress-MP Optimizer MIP solver (version 15.30). In particular the separation algorithm used to generate the (l, S) inequalities (4.13) has been directly coded in Mosel. See `http://www.dashoptimization.com` for more information about this software. All the tests reported here have been carried out on a 1.7 GHz PC (centrino) with 1 GB of RAM running under Windows XP.

Apart from the multi-commodity reformulation, the reformulations of the single-item problems $WW\text{-}U$ and $LS\text{-}U$ used here are studied in detail in Chapter 7. The $WW\text{-}CC$ reformulation is studied in Chapter 9. Appropriate references to these results are given in these chapters.

An introduction to the techniques used to prove that some valid inequalities suffice to describe the convex hull of solutions to a model is given in Section 6.4. For a general presentation of the various techniques that can be used to prove that some valid inequalities are facet defining, and for related topics, we refer the reader to Nemhauser and Wolsey [126].

Springer

http://www.springer.com/978-0-387-29959-4

Production Planning by Mixed Integer Programming
Pochet, Y.; Wolsey, L.A.
2006, XXIV, 500 p., Hardcover
ISBN: 978-0-387-29959-4